On voit des %xx dans certaines URLs, qu'est-ce que c'est ?

Ces caractères viennent de la présence de caractères « interdits » dans les noms de fichier — en fait, interdits aux débutants. Les choses sont plus complexes : ces caractères ne sont pas forcément interdits dans les *noms de fichiers*, mais ils le sont *dans les URLs* qu'on place dans les liens pour appeler ces fichiers. Une URL a une syntaxe très stricte dans laquelle les caractères

; / ? : @ & = + $,

sont réservés à des usages particuliers, dépendant de l'endroit où on se trouve dans l'URL. D'autres caractères sont interdits ; seuls les caractères alphanumériques majuscules et minuscules et les signes de ponctuation suivants

- _ . ! ~ * ' ( )

sont permis. *L'espace et les caractères accentués notamment font partie des caractères interdits.*

Il arrive cependant qu'on veuille mettre des caractères interdits ou réservés dans une URL. Il est bien entendu très déconseillé de mettre de tels caractères dans des noms de fichier, mais on peut parfois passer outre. Surtout, on peut être amené à prolonger les adresses de fichiers par des *chaines de requête* que la page appelée va déchiffrer pour exécuter différents scripts. Notre méthode pour passer des variables javascript d'une page à l'autre en est un exemple. La transmission du contenu d'un formulaire au script cgi de wanadoo qui va le traiter repose également sur une requête de ce genre. On peut donc être amené à mettre dans une URL des caractères qui ne devraient pas s'y trouver. On les alors obligé de les camoufler d'une manière spéciale ; on dit qu'il faut les « échapper » ou « %-encoder ». ou encore, les « url-encoder ».

Pour simplifier (parce que ça peut être plus compliqué), nous en resterons à ce schéma où une URL se compose d'abord d'un *chemin* (adresse d'un fichier ou d'un programme sur le serveur), puis d'une *chaine de requête* facultative, séparée du chemin par un « ? ». Faites une recherche sur un moteur de recherche quelconque ; vous verrez que l'URL de la page de résultat a bien cette tête-là.

Le %-encodage sera identique dans les deux parties de l'URL, à ceci près que les caractères **: @ & = + , $** ne sont réservés que dans la chaine de requête, pas dans le chemin. Ils pourront donc figurer tels quels dans ce chemin.

La règle est simple : on remplace le caractère par un «%» suivi de son code hexadécimal dans l'encodage courant du document (c'est-à-dire la norme iso-8859-1 en général pour l'Europe de l'Ouest et les Amériques). La table ci-dessous donne les plus importants d'entre eux pour nous autres français (cliquez ici pour obtenir une table complète).

espace	à	â	ç	è	é	ê	î	ô	ù	û	;	/	?	:	=	+
20	e0	e2	e7	e8	e9	ea	ee	f4	f9	fb	3b	2f	3f	3a	3d	2b

Par exemple, le code de l'espace étant 20 et celui de « é » étant e9 (ou E9, car on peut utiliser indifféremment des majuscules ou des minuscules pour ces chiffres hexadécimaux), on pourrait rencontrer des URLs comme http://perso.wanadoo.fr/coin.des.experts/nom%20accentu%E9.html pour charger un fichier "*nom accentué.html*".

Le «?» servant de séparateur ne doit évidemment pas être %-encodé; par contre, les autres «?» ne peuvent figurer tels quels ni dans le chemin ni dans la chaîne de requête et il doivent donc alors être %-encodés en%3F (ou %3f)

Le signe «+» est un caractère réservé dans la chaîne de requête. Non encodé, il représente un espace, c'est-à-dire que avec+espace et avec%20espace se décodent de la même manière en «avec espace». Par contre, si on veut vraiment un symbole «+» dans la chaine de requête, il faut l'encoder en %2b ou %2B. Nous ne nous étendrons pas sur la signification des autres caractères réservés.

Les règles ci-dessus vous donnent ainsi les règles essentielles pour faire des lien vers des fichiers dont les noms comporteraient des accents ou d'autres caractères exotiques et qui auraient été transmis tels quels au serveur. Nous persistons néammoins à vous déconseiller d'utiliser ces caractères, pour plusieurs raisons:

- ces URLs encodées ne sont pas particulièrement faciles à lire et à corriger en cas d'erreur de frappe (surtout si on fait l'encodage à la main)
- l'utilisation de ces caractères dans les noms de fichiers entraine parfois de gros problèmes de lecture des répertoires sur certains serveurs. Avec des programmes FTP très répandus, comme *CuteFTP*, il y a encore un an, une partie des fichiers pouvait devenir invisible sur Wanadoo.
- d'autres programmes FTP peuvent faire cet encodage sur les *noms de fichiers* eux-même au moment du transfert, selon leurs réglages de préférence, ce qui provoque alors des cassures de liens (car il ne faut évidemment encoder *que* les URLs)
- le fonctionnement en local n'est pas garanti (surtout avec les navigateurs sur MacOS), ce qui est assez fâcheux pour la mise au point du site

Pour la partie « chaine de requête », le problème se pose généralement dans des scripts javascripts et il existe alors une fonction escape() qui fait l'encodage automatiquement.

Documentation de référence

- Uniform Resource Identifiers : http://www.ietf.org/rfc/rfc2396.txt
- HTTP/1.1 : http://www.ietf.org/rfc/rfc2616.txt
- HTML : http://www.w3.org/TR/html4/appendix/notes.html#h-B.2

(TOM)